

Quick Reference for vi

This quick reference lists commands you can use in the **vi** editor on Hewlett-Packard's UNIX™ System, HP-UX.

Notations

- Commands beginning with **:** (colon) must end with **Return**. These commands represent escape commands to the **ex** editor. You can reference the **ex** tutorial for more details.
- *file* is the name of file
- *cursor_cmd* is a cursor movement command (e.g., **G j w b**)
- *char* is a single character
- *str* is a character string (can contain pattern matching characters)
- **CTRL-x** means you press **CTRL**, hold it down, and press the **x** key.
- *n,m* can be two line numbers (e.g., 4, 50), line marker (e.g., ., \$), or search expression (e.g., /string1/, /string2/).
- (*a-z*) means you choose a letter from *a* through *z*

Modes

Command Mode	When you are not inserting or changing text, you can move the cursor and run commands (e.g., searching, deleting, saving). Pay attention to the case of the commands; check the Caps lock key if vi behaves strangely.
Insert Mode	When you insert or change more than one character of text, you cannot use command mode commands. To leave the insert mode, press ESC .

Start a vi Session

vi <i>file</i>	Edit <i>file</i>
vi -r <i>file</i>	Edit last saved version of <i>file</i> after system or editor crash
vi + n <i>file</i>	Edit <i>file</i> and place cursor at line <i>n</i>
vi + <i>file</i>	Edit <i>file</i> and place cursor on last line
vi <i>file1 file2 ...filen</i>	Edit <i>file1</i> through <i>filen</i> ; After saving changes in <i>file1</i> , enter :n for next file
vi +/str <i>file</i>	Edit <i>file</i> and place cursor at line containing <i>str</i>

Save Text and Exit vi

ZZ or :wq or :x	Save file and exit vi
:w <i>file</i>	Save <i>file</i> but do not exit; omitting <i>file</i> saves current file
:w! <i>file</i>	Save <i>file</i> overriding normal checking
:n,mw <i>file</i>	Write lines <i>n</i> through <i>m</i> to <i>file</i>
:n,mw>> <i>file</i>	Append lines <i>n</i> through <i>m</i> to end of <i>file</i>
:q	Leave vi, saving changes before last write (you may be prompted to save first)
:q!	Leave vi without saving any changes since last write
Q	Escape vi into ex editor with same file; :vi returns.
:e!	Re-edit current file, disregarding changes since last write

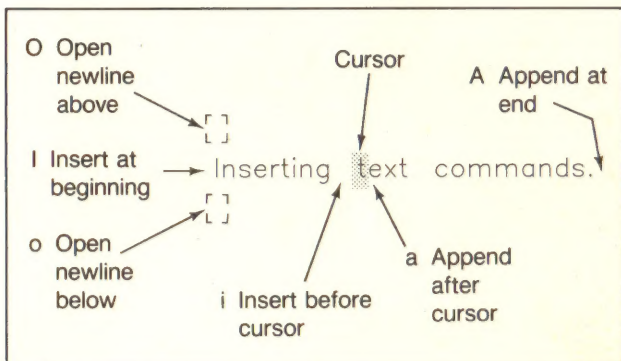
Status Commands

:.=	Print current line number
:=	Print number of lines in file
CTRL-q	Show file name, current line number, total lines in file, and percent of file location
:l (letter "l")	Display tab (^l) backslash (\) backspace (^H) newline (\$) bell (^G) formfeed (^L) of current line in status line

Inserting Text

To leave the insert mode, press **[ESC]**.

a	Append after cursor
A	Append after end of current line
i	Insert before cursor
I	Insert before beginning of current line
o	Open new line below current line and insert
O	Open new line above current line and insert
[CTRL]-[v] char	While inserting, ignore special meaning of <i>char</i> (e.g., for inserting characters like [ESC] and control characters)
:r file	Read <i>file</i> , and insert after current line
:nr file	Read <i>file</i> , and insert after line <i>n</i>

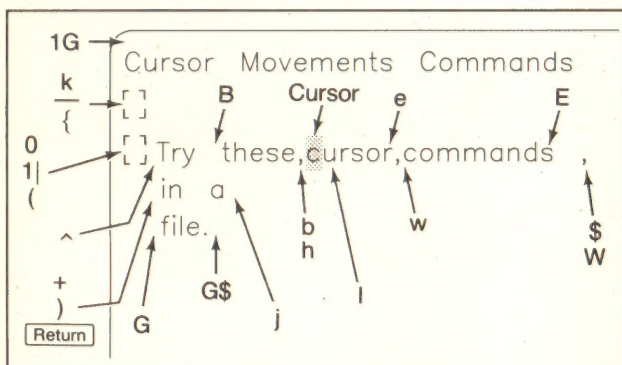


Undoing and Repeating Commands

u	Undo last command
U	Restore current line to original state
"np	Retrieve last <i>n</i> th delete (last 9 deletes are in a buffer)
"1pu.u.	Scroll through the delete buffer until you retrieve desired delete (repeat u.)
n	Repeat last / or ? search command
N	Repeat, in reverse direction, last / or ? search command
; (semi-colon)	Repeat last f F t or T search command
, (comma)	Repeat, in reverse direction, last f F t or T search command
. (period)	Repeat last text change command

Moving the Cursor

k or CTRL-p	Up
j or CTRL-j or CTRL-n	Down
h or CTRL-h or Back space	Left
l or Space	Right
w or W	Start of next word; W ignores punctuation
b or B	Start of previous word; B ignores punctuation
e or E	End of next word; E ignores punctuation
0 (zero) or	First column in current line
n	Column <i>n</i> in current line
^ (caret)	First non-blank character in current line
\$	Last character in current line
+ or Return	First character in next line
-	First non-blank character in previous line
1G	First line in file
G	Last line in file
G\$	Last character in file
nG	Line <i>n</i> in file
(Back to beginning of sentence
)	Forward to beginning of next sentence
{	Back to beginning of paragraph
}	Forward to beginning of next paragraph



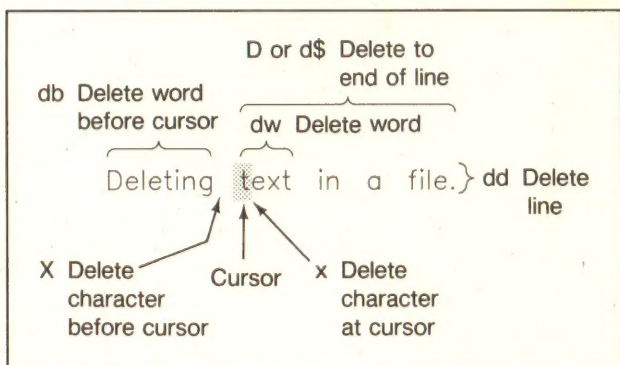
Section Positioning

Mark sections by placing { in first column.

[[Back to beginning of section
]]	Forward to beginning of next section

Deleting Text

CTRL h or Back space	While inserting, delete previous character
CTRL w	While inserting, delete previous word
CTRL x	While inserting, delete to start of inserted text
nx	Delete <i>n</i> characters beginning with current; omitting <i>n</i> deletes current character
nX	Delete previous <i>n</i> characters; omitting <i>n</i> deletes previous character
xp	Switch character at cursor with following character
ndw	Delete next <i>n</i> words beginning with current; omitting <i>n</i> deletes current word
ndb	Delete previous <i>n</i> words; omitting <i>n</i> deletes previous word
ndd	Delete <i>n</i> lines beginning with current; omitting <i>n</i> deletes current line
:n,m	Delete lines <i>n</i> through <i>m</i>
D or d\$	Delete from cursor to end of current line
d <i>cursor_cmd</i>	Delete text to <i>cursor_cmd</i> (e.g., dG deletes from current line to end of file)



Placing Marks in the Text

m(a-z)	Mark current position with a letter a through z (e.g., ma)
'(a-z)	Move cursor to position (a-z) (e.g., 'a)
' ' or ' ' (single quotes or grave accents)	Move cursor to location before last / ? or G command

Pattern Matching

Pattern matching characters help find strings with similar characteristics.

<code>:set magic</code>	Allow pattern matching with special characters (default)
<code>:set nomagic</code>	Allow only <code>^</code> and <code>\$</code> as special characters
<code>^</code> (caret)	Match beginning of line
<code>\$</code>	Match end of line
<code>.</code>	Match any single character
<code>\<</code>	Match beginning of word
<code>\></code>	Match end of word
<code>[str]</code>	Match any single character in <i>str</i>
<code>[^str]</code>	Match any character not in <i>str</i>
<code>[a-n]</code>	Match any character between <i>a</i> and <i>n</i>
<code>*</code>	Match zero or more occurrences of previous character in expression
<code>\</code>	Escape meaning of next character (e.g., <code>\\$</code> lets you search for <code>\$</code>)
<code>\\</code>	Escape the <code>\</code> character

Indenting Text

<code>CTRL-i</code> or <code>Tab</code>	While inserting, insert one shift width
<code>:set ai</code>	Turn on auto-indentation
<code>:set sw=<i>n</i></code>	Set shift width to <i>n</i> characters
<code><i>n</i><< or <i>n</i>>></code>	Shift <i>n</i> lines left or right (respectively) by one shift width; omitting <i>n</i> shifts one line
<code>< or ></code>	Use with cursor command to shift multiple lines left or right



HEWLETT
PACKARD

HP Part Number
98597-90000

Microfiche No. 98597-99000
Printed in U.S.A. 9/87



98597-90630

For Internal Use Only

Searching

%	Search to beginning of balancing () [] or { }
fchar	Search forward in current line to <i>char</i>
Fchar	Search backward in current line to <i>char</i>
tchar	Search forward in current line to character before <i>char</i>
Tchar	Search backward in current line to character after <i>char</i>
/str <input type="button" value="Return"/>	Find <i>str</i>
?str <input type="button" value="Return"/>	Search in reverse for <i>str</i>
:set ic	Ignore case when searching
:set noic	Pay attention to case when searching (default)

Global Search and Replace

:n,ms/str1/str2/opt	Search from <i>n</i> to <i>m</i> for <i>str1</i> . Replace <i>str1</i> with <i>str2</i> , using <i>opt</i> . <i>opt</i> can be g for global change, c to confirm change (press y to acknowledge, <input type="button" value="Return"/> to suppress), and p to print changed lines.
&	Repeat last :s command
:g/str/cmd	Run <i>cmd</i> on all lines that contain <i>str</i>
:g/str1/s/str2/str3/	Find line containing <i>str1</i> , replace <i>str2</i> with <i>str3</i>
:v/str/cmd	Execute <i>cmd</i> on all lines that do not match <i>str</i>

Copying and Placing Text

nyy or nY	Yank <i>n</i> lines (place in buffer); omitting <i>n</i> yanks current line
ycursor_cmd	Yank from cursor to <i>cursor_cmd</i> (e.g., yG yanks current line to last line in file)
"(a-z)nyy or "(a-z)ndd	Copy or delete <i>n</i> lines into named buffer <i>a</i> through <i>z</i> ; omit <i>n</i> for current line
p (lower-case)	Put yanked text after cursor (print buffer); also prints last deleted text
P	Put yanked text before cursor; also prints last deleted text
"(a-z)p or "(a-z)P	Put lines from named buffer <i>a</i> through <i>z</i> after or before current line

Changing Text

Preceding these commands with *n* (a number) repeats the command *n* times.

<i>rchar</i>	Replace current character with <i>char</i>
<i>Rtext</i> [ESC]	Replace current character(s) with <i>text</i>
<i>s</i> <i>text</i> [ESC]	Substitute <i>text</i> for current character
<i>S</i> or <i>cc text</i> [ESC]	Substitute <i>text</i> for entire line
<i>cwtext</i> [ESC]	Change current word to <i>text</i>
<i>Ctext</i> [ESC]	Change rest of current line to <i>text</i>
<i>ccursor_cmd text</i> [ESC]	Change to <i>text</i> from current position to <i>cursor_cmd</i>

Joining Lines

<i>J</i>	Join next line to end of current line
<i>nJ</i>	Join next <i>n</i> lines

Cursor Placement and Adjusting the Screen

<i>H</i>	Move cursor to top line of screen
<i>nH</i>	Move cursor to line <i>n</i> from top of screen
<i>M</i>	Move cursor to middle of screen
<i>L</i>	Move cursor to bottom line of screen
<i>nL</i>	Move cursor to Line <i>n</i> from bottom of screen
[CTRL] [e]	Move screen up one line
[CTRL] [y]	Move screen down one line
[CTRL] [u]	Move screen up 1/2 page
[CTRL] [d]	Move screen down 1/2 page
[CTRL] [b]	Move screen up one page
[CTRL] [f]	Move screen down one page
[CTRL] [I] (letter "I")	Redraw screen
<i>z</i> [Return]	Make current line top line on screen
<i>nz</i> [Return]	Make line <i>n</i> top line on screen
<i>z.</i>	Make current line middle line
<i>nz.</i>	Make line <i>n</i> middle line on screen
<i>z-</i>	Make current line bottom line
<i>nz-</i>	Make line <i>n</i> bottom line on screen

Shell Escape Commands

: ! <i>cmd</i>	Execute shell command <i>cmd</i> ; you can add these special characters to indicate: % name of the current file # name of last file edited
: !!	Execute last shell command
: r! <i>cmd</i>	Read and insert output from <i>cmd</i>
: f <i>file</i>	Rename current file to <i>file</i>
: w ! <i>cmd</i>	Send currently edited file to <i>cmd</i> as standard input and execute <i>cmd</i>
: cd <i>dir</i>	Change the current working directory to <i>dir</i> (\$HOME is default)
: sh	Start a sub-shell (CTRL-d returns to editor)
: so <i>file</i>	Read and execute commands in <i>file</i> (<i>file</i> is a shell script)

Shell Filters

! <i>cursor_cmd cmd</i>	Send text from current position to <i>cursor_cmd</i> to shell command <i>cmd</i> . Replace original text in file with output from <i>cmd</i>
! } sort Return	Example: Sort from current position to end of paragraph and replace text with sorted text

Macros and Abbreviations

: map <i>key cmd_seq</i>	Define <i>key</i> to run <i>cmd_seq</i> when pressed
: map	Display all created macros on status line
: unmap <i>key</i>	Remove macro definition for <i>key</i>
: ab <i>str string</i>	When <i>str</i> is inserted, replace with <i>string</i>
: ab	Display all abbreviations
: una <i>str</i>	Unabbreviate <i>str</i>

Map allows you to define strings of vi commands. Place in .exrc to run each time you enter vi. For long macros, set the `notimeout` option. If you embed control characters (e.g., keys like `ESC`) in the macro, you need to precede them with `CTRL-v`. If you need to include quotes ("), precede them with \ (backslash). Unused keys in vi are: K V g q v * = and the function keys.

Example:

```
:map v /I CTRL-v ESC dwIYou CTRL-v ESC ESC
```

When v is pressed, search for "I" (/I `ESC`), delete word (dw) and insert "You" (iYou `ESC`). `CTRL-v` allows `ESC` to be inserted.

Setting Options

Options shown here are default. To change them, either set them (`:set option`) or unset them (`:set nooption`). To run options each time you enter vi, place in `.exrc` file in home directory and omit preceding colons (:).

<code>:set all</code>	Print all options
<code>:set nooption</code>	Turn off <i>option</i>
<code>:set noai</code>	Set automatic indentation
<code>:set ap</code>	Print line after <code>d c J m : s t u</code> command
<code>:set bf</code>	Discard control characters from input
<code>:set eb</code>	Precede error messages with bell
<code>:set noic</code>	Ignore case when searching
<code>:set dir=tmp</code>	Set directory of buffer file
<code>:set lisp</code>	Modify brackets for Lisp compatibility
<code>:set magic</code>	Pattern match with special characters
<code>:set mesg</code>	Allow other users to send messages
<code>:set nolist</code>	Show tabs (^1) and end of line (\$)
<code>:set nonu</code>	Prefix lines with line number
<code>:set opt</code>	Speed output: eliminate automatic Return
<code>:set prompt</code>	Prompt for command mode input with :
<code>:set nore</code>	Simulate smart terminal on dumb terminal
<code>:set remap</code>	Allow macros within macros
<code>:set noreport</code>	Indicate largest size of changes reported on status line
<code>:set ro</code>	Change file to read only
<code>:set scroll=n</code>	Set <i>n</i> lines for CTRL - d and <code>z</code>
<code>:set sh=shell_path</code>	Set shell escape (default is <code>/bin/sh</code>)
<code>:set showmode</code>	Indicate input or replace mode
<code>:set sw=8</code>	Set the shift width
<code>:set term</code>	Print terminal type
<code>:set noterse</code>	Shorten error messages with <code>terse</code>
<code>:set notimeout</code>	Eliminate one second time limit for macros
<code>:set tl=0</code>	Set significance of tags beyond this many characters (0 means all)
<code>:set ts=8</code>	Set tab stops for text input
<code>:set nowa</code>	Inhibit normal checks before write commands
<code>:set warn</code>	Warn "No write since last change"
<code>:set window=n</code>	Set number of lines in a text window
<code>:set wm=n</code>	Set automatic wraparound <i>n</i> spaces from right margin (e.g., <code>:set wm=8</code>)